Ajax

Day01

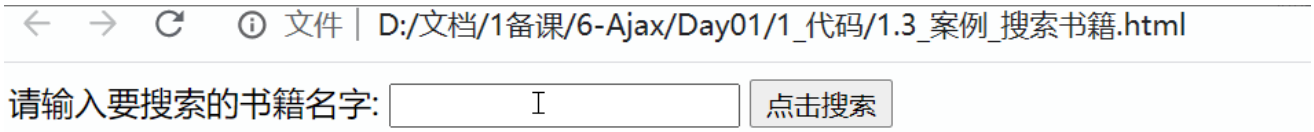
课上练习

暂无

案例

1.3 案例 - 输入书名 - 搜索相关信息

效果演示



模板标签

```
<!DOCTYPE html>
<html lang="en">

    <head>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
        <title>案例_搜索书籍</title>
    </head>

    <body>
        <div>
            <span>请输入要搜索的书籍名字:</span>
            <input type="text" id="bookName">
            <button id="btn">点击搜索</button>
        </div>
        <div id="result">

        </div>
        <script src="./js/jquery3.5.1.js"></script>
    </body>
```

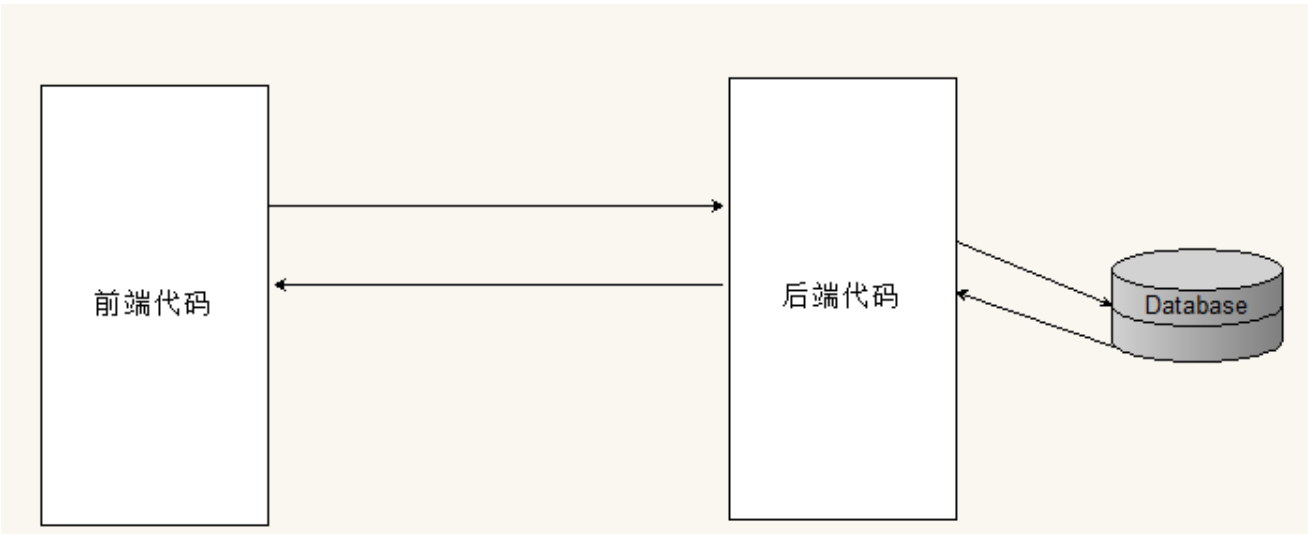
```
</html>
```

正确的js代码

```
// 1. 按钮 - 点击事件
$("#btn").on("click", function () {
    // 2. 收集输入框的值 - 通过$.get()发给url并把值一起发给后台
    $.get('http://123.57.109.30:3006/api/getbooks', {
        bookname: $("#bookName").val()
    }, function (res) { // 后台返回结果res
        if (res.data.length == 0) {
            $("#result").html("查无此书");
        } else {
            var arr = res.data;
            var str = ``;
            arr.forEach(function (obj) {
                let { author, publisher, bookname } = obj;
                str += `<p><span>作者: ${author}</span>  <span>出版社: ${publisher}</span> <span>书籍名: ${bookname}</span></p>`
            })
            $("#result").html(str);
        }
    })
})
```

2. 案例 - 图书管理

- 前端通过Ajax发送请求（调用接口）给服务器
- 服务器处理请求，根据请求的路径再操作数据库（读取数据，添加、删除或者修改数据）
- 数据库的操作结果再通过后端返回给前端



- 增删改查 CRUD
    - create 创建
    - read 读取

    - update 更新
    - delete 删除
- Bootstrap 界面布局采用 - 布局直接使用, 重点练习JS操作页面 - Ajax和后台交互
- 图书管理案例效果图
    - 查询
    - 添加
    - 删除

模板标签

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>图书管理</title>
    <link rel="stylesheet" href="./lib/bootstrap.css" />
</head>

<body style="padding: 15px;">
    <!-- 添加图书的Panel面板 -->
    <div class="panel panel-primary">
        <div class="panel-heading">
            <h3 class="panel-title">添加新图书</h3>
        </div>
        <div class="panel-body form-inline">

            <div class="input-group">
                <div class="input-group-addon">书名</div>
                <input type="text" class="form-control" id="iptBookname" placeholder="请输入书
名">
            </div>

            <div class="input-group">
                <div class="input-group-addon">作者</div>
                <input type="text" class="form-control" id="iptAuthor" placeholder="请输入作者">
            </div>
```

```
            <div class="input-group">
                <div class="input-group-addon">出版社</div>
                <input type="text" class="form-control" id="iptPublisher" placeholder="请输入出版
社">
            </div>

            <button id="btnAdd" class="btn btn-primary">添加</button>

        </div>
    </div>


    <!-- 图书的表格 -->
    <table class="table table-bordered table-hover">
        <thead>
            <tr>
                <th>Id</th>
                <th>书名</th>
                <th>作者</th>
                <th>出版社</th>
                <th>操作</th>
            </tr>
        </thead>
        <tbody id="tb">
            <!-- <tr>
        <td>1</td>
        <td>史记</td>
        <td>司马迁</td>
        <td>河北出版社</td>
        <td>
          <a href="javascript:;" class="del">删除</a>
        </td>
      </tr>
      <tr>
        <td>2</td>
        <td>斗破苍穹</td>
        <td>天蚕土豆</td>
        <td>网络
          出版社</td>
        <td>
          <a href="javascript:;" class="del">删除</a>
        </td>
      </tr> -->
        </tbody>
    </table>

    <script src="./lib/jquery.js"></script>

</body>

</html>
```

2.1 案例 - 图书管理 - 初始化页面

这里把自己的appkey带上, 可以获取专属自己的数据

```
// 请求接口，获取所有的图书
function getBooks() {
    // 1.1 使用Ajax - 把数据请求回来
    $.ajax({
        url: 'http://123.57.109.30:3006/api/getbooks',
        data: {
            appkey: '7250d3eb-18e1-41bc-8bb2-11483665535a'
        },
        type: 'get',
        success(res) {
            // 1.2 判断状态码 - 统一状态是200代表成功
            if (res.status == 200) {
                // 1.3 提取数据 - 铺设页面
                $(res.data).each(function (index, obj) {
                    $("#tb").prepend($(`<tr>
<td>${obj.id}</td>
<td>${obj.bookname}</td>
<td>${obj.author}</td>
<td>${obj.publisher}</td>
<td>
<a href="javascript:;" class="del" data-id="${obj.id}">删除</a>
</td>
</tr>`))
                })
            } else { // 1.4 如果不是200状态，直接打印后台返回的错误提示
                alert(res.msg);
            }
        }
    });
}

// 1.5 网页打开 - 到这里直接调用
getBooks();
```

2.2 案例 - 图书管理 - 新增图书

```
// 2.1 写一段代码，请求接口，添加图书
$('#btnAdd').click(function () {
    // 2.2 获取输入框的值（三个）
    var bookname = $('#iptBookname').val().trim();
    var author = $('#iptAuthor').val().trim();
    var publisher = $('#iptPublisher').val().trim();
    // 2.3 判断不能为空，如果有一个为空，提示并终止代码继续执行
    if (bookname == '' || author == '' || publisher == '') {
        return alert('内容不能为空');
    }
    // 2.4 按照接口文档，调用添加书籍的接口，完成添加

    $.ajax({
```

```
        type: 'POST',
        url: 'http://123.57.109.30:3006/api/addbook',
        data: {
            bookname: bookname,
            author: author,
            publisher: publisher,
            appkey: '7250d3eb-18e1-41bc-8bb2-11483665535a'
        },
        success: function (res) {
            if (res.status === 201) {
                // 2.5 添加成功
                // 添加成功的话，以前todolist案例-重新加载一遍是为了保证索引值能重新对上分配，而现在每个商品有自己的唯一性的id，所以就不用在重新铺设了(保证性能-所以以后基本能用id用id)
                var obj = res.data; // 接受后台返回插入成功的这个对象(带后台生成的id的)
                $("#tb").prepend($(`<tr>
<td>${obj.id}</td>
<td>${obj.bookname}</td>
<td>${obj.author}</td>
<td>${obj.publisher}</td>
<td>
<a href="javascript:;" class="del" data-id="${obj.id}">删除</a>
</td>
</tr>`))
            } else {
                // 添加失败
                alert('添加失败');
            }
        }
    });

});
```

2.3 案例 - 图书管理 - 删除图书

```
// 3.1 因为a是动态创建的，所以在这里要执行 - 只能用事件委托，不然得写在getBoods里面的下面 再绑定，代码写在一起不好
$('#tb').on('click', '.del', function () {
    // 3.2 对于敏感操作，最好给一个提示（询问是否要删除）
    if (!confirm('你确定要删除吗，你好狠！')) {
        return;
    }
    // 3.3 如果确定是删除，按照接口，完成接口的要求
    var id = $(this).attr('data-id');
    $.get('http://123.57.109.30:3006/api/delbook', { id: id, appkey: '7250d3eb-18e1-41bc-8bb2-11483665535a' }, res => {
        // 不用判断，先给出提示(因为，无论成功还是失败，都要给出提示)
        alert(res.msg);
        // 判断，如果成功了，移除掉当前的这一行
        if (res.status === 200) {
            // 移除当前这一行
            $(this).parents('tr').remove();;

        }
```

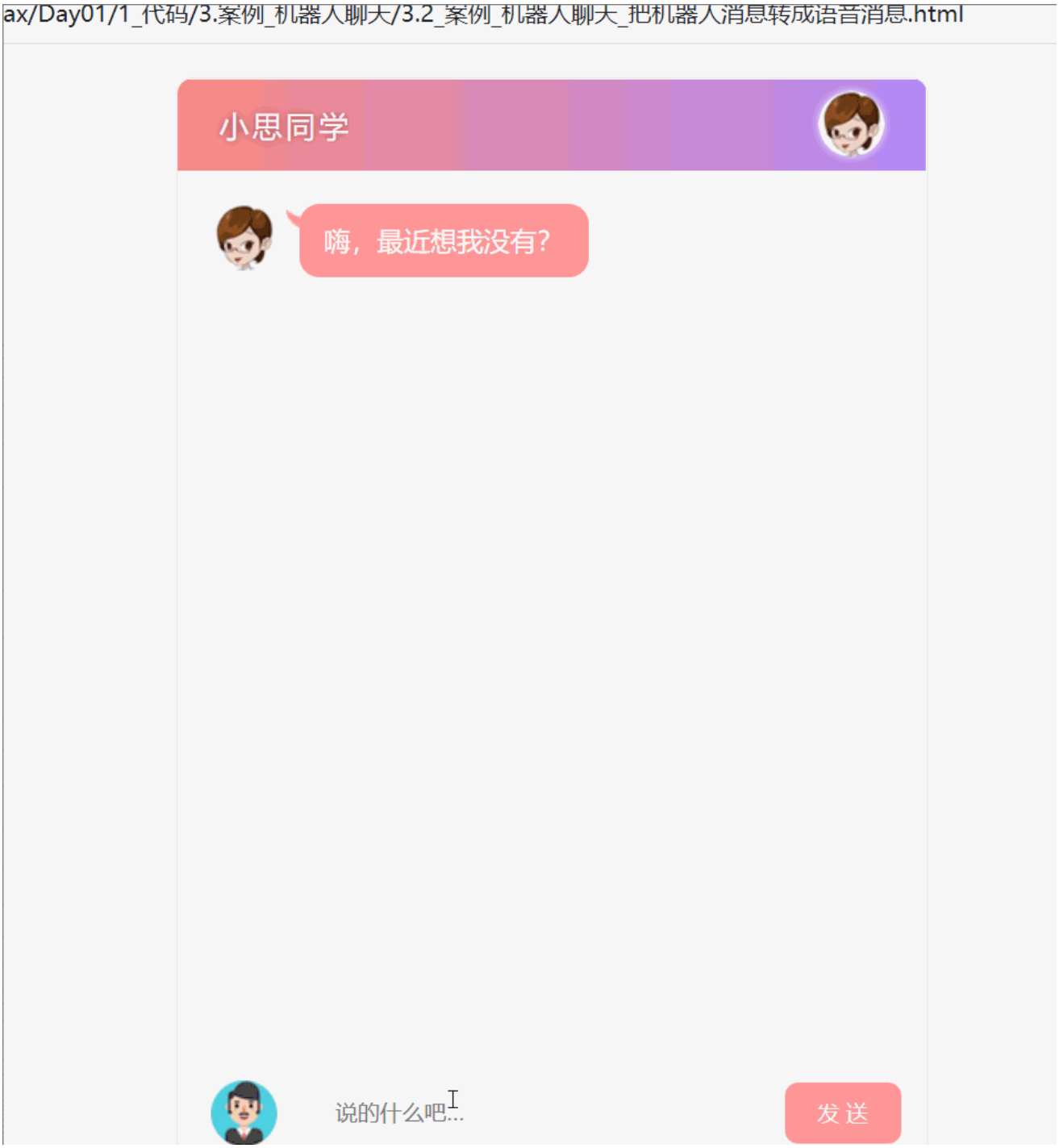
```
    });
});
```

3. 案例 - 聊天机器人

静态标签和样式请在images/素材中看序号查找

需求是调用接口得到机器人返回的信息

效果演示:



3.1 案例 - 机器人 - 发送消息和获取消息

```
$(function () {
```

```
    // 1.1 发送按钮 - 点击事件
    $('#btnSend').on('click', function () {
        // 1.2 输入内容 - 为空阻止往下执行
        var text = $('#ipt').val().trim()
        if (text.length <= 0) {
            return; // 空内容阻止发送
        }
        // 1.3 如果用户输入了聊天内容，则将聊天内容追加到页面上显示
        $('#talk_list').append('<li class="right_word"><img src="img/person02.png" /> <span>' +
text + '</span></li>')
        $('#ipt').val('') // 清空输入框
        // 重置滚动条的位置
        resetui()
        // 1.4 发起请求，获取聊天内容
        getMsg(text)
    })

    // 1.5 获取聊天机器人发送回来的消息
    function getMsg(text) {
        // 1.6 调用自己后台的接口
        $.ajax({
            type: 'GET',
            url: 'http://123.57.109.30:3006/api/robot',
            data: {
                spoken: text // 把我说的内容发给后台
            },
            success: function (res) {
                // 1.7 判断获取正确 - 把内容显示在左侧
                if (res.message === 'success') {
                    // 接收聊天消息
                    var msg = res.data.info.text
                    $('#talk_list').append('<li class="left_word"><img src="img/person01.png" />
<span>' + msg + '</span></li>')
                    // 重置滚动条的位置
                    resetui()
                }
            }
        })
    }

    // 1.8 为文本框绑定 keyup 事件
    // 当用户按下键盘回车，也是要发送消息
    $('#ipt').on('keyup', function (e) {
        if (e.keyCode === 13) {
            $('#btnSend').click()
        }
    })
})
```

3.2 案例 - 机器人 - 消息转为语音

```
<!-- 注意: 只要为 audio 指定了新的 src 属性, 而且指定了 autoplay, 那么, 语音就会自动播放 -->
<audio src="" id="voice" autoplay style="display: none;"></audio>
```

```
// 2.0 把文字转化为语音进行播放
function getVoice(text) {
    // 2.1 语音文件,  还是要Ajax调用接口请求 (注意这次拿回来的是语音的url, 不是字符串数据了)
    $.ajax({
        type: 'GET',
        url: 'http://123.57.109.30:3006/api/synthesize',
        data: {
            text: text
        },
        success: function (res) {
            if (res.status === 200) {
                // 2.2 把语音消息的url, 设置给audio标签 - 播放语音
                $('#voice').attr('src', res.voiceUrl)
            }
        }
    })
}
```

作业

暂无

Day02

课上练习

练习1: 编写原生Ajax代码 - 请查询公共数据(无需带appkey参数), 中id为有的数据信息 (这是要哪数据吧? 用那种请求方式呢?)

练习2: 编写原生Ajax代码 - 请向自己的表中(带appkey和自己的值), 向后台随便增加一条书籍信息, 然后调用使用vscode的.http文件插件, 查询自己名下的书籍信息的接口, 查看是否添加成功 - (这是要发送数据吧? 应该用什么请求方式呢?)

js正确代码:

```
// 练习1: 编写原生Ajax代码 - 请查询公共数据(无需带appkey参数), 中id为有的数据信息 (这是要哪数据吧? 用那种请求方式呢?)
let ajaxObj = new XMLHttpRequest();
ajaxObj.onreadystatechange = function(){
    if (ajaxObj.readyState == 4) {
        if (ajaxObj.status == 200) {
            console.log(ajaxObj.responseText);
        }
    }

}
```

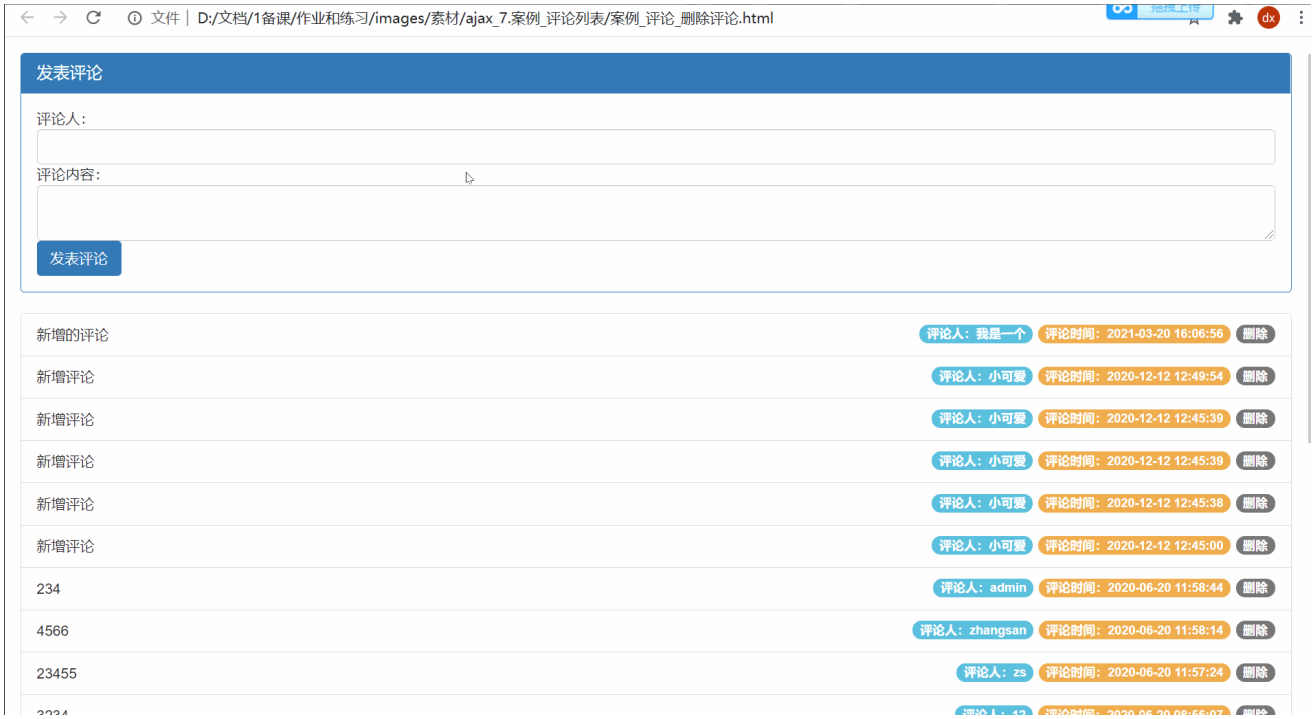
```
// 总结：多个参数要在?后面 以参数名=值&参数名=值
ajaxObj.open("GET", "http://123.57.109.30:3006/api/getbooks?id=369");
ajaxObj.send();

// 练习2：编写原生Ajax代码 - 请向自己的表中(带appkey和自己的值)，向后台随便增加一条书籍信息，然后调用使用vscode的.http文件插件，查询自己名下的书籍信息的接口，查看是否添加成功 -（这是要发送数据吧？应该用什么请求方式呢?)
// 1. 实例化ajax对象
let ajaxObj = new XMLHttpRequest();
// 2. 绑定事件
ajaxObj.onreadystatechange = function(){
    // 5. 接受返回的结果
    if (ajaxObj.readyState == 4) {
        if (ajaxObj.status == 200) {
            console.log(ajaxObj.responseText);
        }
    }
}
// 3. 设置参数
ajaxObj.open("POST", "http://123.57.109.30:3006/api/addbook");

// (新) - 设置内容的类型
ajaxObj.setRequestHeader("Content-Type", "application/x-www-form-urlencoded");

// 4. 发送请求
// (新)-post的参数要写在send()里面
ajaxObj.send("bookname=我是vscode插件的书籍&author=vscode&publisher=121&appkey=7250d3eb-18e1-41bc-8bb2-11483665535a");
```

案例

7. 案例 - 评论

标签和样式在images/素材/看序号查找

效果如下:



7.0 - 案例 - 评论 - 初始化列表

```
// 1.0 声明变量 - 定义获取当前页面评论数据 的方法
var nowPage = 1; // 当前要第几页数据
var allPage; // 一共几页
// 1.1 获取铺设评论列表区
function getCommentList() {
    // 1.2 调用接口 - 传递相应参数给后台 - 获取数据
    $.ajax({
        method: 'GET',
        url: `http://123.57.109.30:3006/api/cmtlist?page=${nowPage}&appkey=7250d3eb-18e1-41bc-
8bb2-11483665535a`,
        success: function (res) {
            // 1.3 判断不是200证明报错 - 提示消息
            if (res.status !== 200) return alert(res.msg)
            // 1.4 保存当前一共有多少页
            allPage = res.allPage;

            // 1.5 先清空当前列表页面 - 重新铺设
            $('#cmt-list').empty();
            // 1.6 遍历铺设这次获取回来的数据
            $(res.data).each(function (i, item) {
                $('#cmt-list').append($(`<li class="list-group-item">
<span class="badge del" style="cursor:pointer" theid=${item.id}>删除</span>
<span class="badge" style="background-color: #F0AD4E;">评论时间: ${item.time}</span>
<span class="badge" style="background-color: #5BC0DE;">评论人: ${item.username}</span>
${item.content}</li>`));
            })

            // 1.7 设置页码
            $("#pageShow").html(nowPage);

        }
```

```
    })
}

// 1.8 获取此页的数据
getCommentList();

// 1.9 绑定上一个按钮/下一个按钮点击事件
$("#last").on("click", function () {
    if (nowPage > 1) nowPage--;
    getCommentList()
})
$("#next").on("click", function () {
    if (nowPage < allPage) nowPage++;
    getCommentList()
})
```

7.1 - 案例 - 评论 - 新增评论

```
// 2.0 点击发布事件
$(function () {
    $('#formAddCmt').submit(function (e) {
        e.preventDefault()
        // 2.1 获取表单上的内容形成key=value&key=value字符串
        var data = $(this).serialize()
        // 2.2 拼接我的授权码
        data += "&appkey=7250d3eb-18e1-41bc-8bb2-11483665535a";
        // 2.3 post发送数据给后台
        $.post('http://123.57.109.30:3006/api/addcmt', data, function (res) {
            if (res.status !== 201) {
                return alert('发表评论失败！' + res.msg);
            }
            // 2.4 发送成功后，重新获取数据铺设页面
            nowPage = 1;
            getCommentList();
            // 2.5 复位表单输入框
            $('#formAddCmt')[0].reset()
        })
    })
})
```

7.2 - 案例 - 评论 - 删除评论

```
// 3.0 事件委托 - 给所有删除绑定绑定点击事件
$("#cmt-list").on("click", ".del", function () {
    // 3.1 去上面给删除标签绑定数据对应id，点击获取评论对应唯一性的id
    var theId = $(this).attr("theid");
    // 3.2 调用删除接口 - 让后台删除这个数据
    $.ajax({
        method: 'GET',
        url: `http://123.57.109.30:3006/api/delcmt?id=${theId}&appkey=7250d3eb-18e1-41bc-8bb2-
11483665535a`,
        success: function (res) {
```

```
            // 3.3 如果不是200状态码，代表删除失败给个提示，阻止代码继续往下走
            if (res.status !== 200) return alert(res.msg);
            // 3.4 如果上面if进不去就走这里，保存当前最大页码(因为当前页没数据了得自动回上一页)
            allPage = res.allPage;
            if (nowPage > allPage) nowPage = allPage;
            // 3.5 设置页码 - 重新获取数据 - 铺设当前页面
            getCommentList();
        }
    })
})
```

作业

暂无

Day03

课上练习

暂无

案例

11. 案例 - 新闻列表

素材在images/下素材文件夹里看序号

- 案例图示如下
    - 调用后台接口获取新闻列表数据
    - 拉到底部加载更多, 如果没有更多数据(后台会返回是否还有更多), 给出提示(关闭下拉加载更多的方法)

11.0 案例 - 新闻列表 - 获取数据

```
// 1. Ajax请求数据 - 铺设第一页数据

var nowPage = 1;
```

```
function loadNewsList() {
    $.ajax({
        url: "http://123.57.109.30:3006/api/news",
        data: { page: nowPage },
        type: "GET",
        success(res) {
            let arr = res.data;
            arr.forEach(function(obj){
                let {id, title, source, cmtcount, tags, img, time} = obj;
                // 把tags的值拆分出来 -形成3个span标签
                let spanStr = ``;
                tags.split(",").forEach(str => {
                    spanStr += `<span>${str}</span>`;
                })
                var theDiv = `<div class="news-item">
                                <img class="thumb" src="http://123.57.109.30:3006${img}" alt="">
                                <div class="right-box">
                                <h1 class="title">${title}</h1>
                                <div class="tags">
                                ${spanStr}
                                </div>
                                <div class="footer">
                                <div>
                                <span>${source}</span>  
                                <span>${time.replace(/T|Z/g, " ").substring(0, time.length - 5)}
</span>
                                </div>
                                <span>评论数: ${cmtcount}</span>
                                </div>
                                </div>
                                </div>`;
                $("#news-list").append(theDiv);
            })
        }
    })
}
loadNewsList();
```

11.1 案例 - 新闻列表 - 下拉加载更多

```
// 3. 监测网页是否滚动到底部
$(document).on("scroll", function(){
    if ($(document).scrollTop() >= document.documentElement.scrollHeight -
document.documentElement.clientHeight) {
        nowPage++;
        loadNewsList();
    }
})
```

11.2 案例 - 新闻列表 - 节流阀

```
// 3. 监测网页是否滚动到底部
```

```
var timer = null; // 保存定时器
$(document).on("scroll", function () {
    if (timer) return; // 如果有定时器再触发函数就回去不继续执行下面的

    timer = setTimeout(function () {
        if ($(document).scrollTop() >= document.documentElement.scrollHeight -
document.documentElement.clientHeight) {
            nowPage++;
            loadNewsList();
        }

        timer = null;
    }, 300);
})
```

11.3 节流函数

```
// 3. 监测网页是否滚动到底部
$(document).on("scroll", throttle(function () {
    if ($(document).scrollTop() >= document.documentElement.scrollHeight -
document.documentElement.clientHeight) {
        nowPage++;
        loadNewsList();
    }
}, 300))

// 4. 定义节流函数
function throttle(fn, theTime) {
    let timer = null;
    return function () {
        if (timer) return;
        timer = setTimeout(() => {
            fn.call(this, ...arguments);
            timer = null;
        }, theTime);
    }
}
```

作业

暂无

Day04

课上练习

暂无

案例

13. 案例 - 淘宝搜索案例

网页素材和样式在images/素材看序号

- 案例图示如下所示

13.0 案例 - 获取联想菜单

- 绑定输入事件并获取输入的值

```
// 1. 根据输入框的值 - 获取联想菜单
$("#ipt").on("input", function () {
    var kw = $(this).val();
    $.ajax({
        // 请求地址
        url: 'https://suggest.taobao.com/sug?q=' + encodeURI(kw),
        // 返回数据格式
        dataType: 'jsonp',
        // 回调函数
        success: function (data) {
            $("#suggest-list").empty();
            data.result.forEach(function(arr, index){
                var theDiv = `<div class="suggest-item">${arr[0]}</div>`;
                $("#suggest-list").append(theDiv);
            })
            $("#suggest-list").show();
        }
    })
})
```

13.1 防抖使用

定时n秒, 到了才执行逻辑代码, 如果事件又触发了, 清除上一个定时器, 重新创建定时器 **最后执行一次**

```
// 1. 定时器
var timer = null;

// 根据输入框的值 - 获取联想菜单
```

```
$("#ipt").on("input", function () {
    var kw = $(this).val();
    // 2. 触发一次事件，清空定时器 - 重新赋予一个(相当于重新计时)
    clearTimeout(timer);
    // 3. 如果不在触发事件，那么3秒后执行逻辑代码
    timer = setTimeout(function () {
        $.ajax({
            // 请求地址
            url: 'http://123.57.109.30:3006/api/sug?q=' + encodeURI(kw),
            // 返回数据格式
            // 回调函数
            success: function (data) {
                $("#suggest-list").empty();
                data.result.forEach(function (arr, index) {
                    var theDiv = `<div class="suggest-item">${arr[0]}</div>`;
                    $("#suggest-list").append(theDiv);
                })
                $("#suggest-list").show();
            }
        })
    }, 300);
});
```

13.2 防抖函数 - 封装

```
// 定义缓存对象
var obj = {};

// 根据输入框的值 - 获取联想菜单
$("#ipt").on("input", debounce(function () {
    var kw = $(this).val();

    // 判断如果对象里有那个数据，直接使用铺设页面
    if (obj[kw] !== undefined) {
        $("#suggest-list").empty();
        obj[kw].result.forEach(function (arr, index) {
            var theDiv = `<div class="suggest-item">${arr[0]}</div>`;
            $("#suggest-list").append(theDiv);
        })
        $("#suggest-list").show();
    } else {
        $.ajax({
            // 请求地址
            url: 'http://123.57.109.30:3006/api/sug?q=' + encodeURI(kw),
            // 返回数据格式
            // 回调函数
            success: function (data) {
                $("#suggest-list").empty();
                obj[kw] = data; // 新的数据保存在逻辑变量obj上
                data.result.forEach(function (arr, index) {
                    var theDiv = `<div class="suggest-item">${arr[0]}</div>`;

                    $("#suggest-list").append(theDiv);
```

```
                })
                $("#suggest-list").show();
            }
        })
    }
}, 300));

// 2. 定义防抖的函数
function debounce(fn, theTime) {
    return function () {
        clearTimeout(fn.timer);
        fn.timer = setTimeout(() => {
            fn.call(this, ...arguments);
        }, theTime);
    }
}
```

13.3 缓存搜索结果

在程序运行期间, 用一个对象变量来保存搜索过的联想菜单关键字 - 下次再输入相同的直接用变量里, 而不用再去请求了

```
// 定时器
var timer = null;
// 1. 定义缓存对象
var obj = {};

// 根据输入框的值 - 获取联想菜单
$("#ipt").on("input", function () {
    var kw = $(this).val();
    clearTimeout(timer);
    timer = setTimeout(function () {
        // 2. 判断如果对象里有那个数据，直接使用铺设页面
        if (obj[kw] !== undefined) {
            $("#suggest-list").empty();
            obj[kw].result.forEach(function (arr, index) {
                var theDiv = `<div class="suggest-item">${arr[0]}</div>`;
                $("#suggest-list").append(theDiv);
            })
            $("#suggest-list").show();
        } else {
            $.ajax({
                // 请求地址
                url: 'http://123.57.109.30:3006/api/sug?q=' + encodeURI(kw),
                // 返回数据格式
                // 回调函数
                success: function (data) {
                    $("#suggest-list").empty();
                    obj[kw] = data; // 3. 新的数据保存在逻辑变量obj上
                    data.result.forEach(function (arr, index) {
                        var theDiv = `<div class="suggest-item">${arr[0]}</div>`;
                        $("#suggest-list").append(theDiv);
                    })
```

```
                    $("#suggest-list").show();
                }
            })
        }
    }, 300);
});
```

作业

暂无